Adaptive 3-Frame Difference 기반 템플릿을 이용한 객체 추적

Object Tracking Using Template Based on Adaptive 3-Frame Difference

김현기[1], 이진형[1], 강지운[1], 조성원[1], 김재민[1], 정선태[2], 장용석[2]

[1] 서울시 마포구 홍익대학교 전기정보제어공학과
E-mail: acehunki@hotmail.com
[2] 서울시 동작구 숭실대학교 전자공학과
E-mail: cst@ssu.ac.kr

요 약

물체를 추적하는데 있어서 추적하고자 하는 물체를 검출하여 템플릿을 만드는 것과 두 물체가 겹쳐지거나 다른 배경에 가려진 물체를 구분하여 추적하는 것은 물체 추적에 있어서 중요한 문제이다. 물체를 검출하여 템플릿을 만드는 방법으로 frame difference를 이용하면 천천히 움직이는 물체를 잘 구분할 수 없는 문제점이 있다. 이를 해결하기 위하여 본 논문에서는 adaptive 3-frame difference를 이용하여 정확한 물체의 템플릿을 생성하는 알고리즘을 제안한다.

Key Words : object tracking, adaptive 3-frame difference, background subtraction, template

1. 서 론

최근 영상을 이용한 물체 추적은 컴퓨터 비전 분야에서 활발하게 연구되는 주제 중 하나이다. 물체 추적의 과정을 크게 구분하면 움직임 검출, 검출된 물체 분류, 물체 추적, 물체의 행동 분석, 다중 채널 카메라에서 들어오는 영상 처리 결과 결합 등으로 나눌 수 있다. 그림 1은 이러한 영상 감시의 흐름을 나타내고 있다 [1].

물체 추적에 있어서 가장 큰 문제점은 두 물체가 겹치거나 겹침 후 분리될 때, 또는 물체가 배경에 가려져 사라진 후에 다시 나타나게 됐을 때 물체를 정확히 판별하지 못하는 것이다. 이를 해결하기 위하여 Template matching을 사용해야 하는데 단순한 배경 분리 (Background Subtraction) 방법을 이용하면 조도의 변화에 민감하기 때문에 본 논문에서는 Adaptive 3-Frame Difference를 이용하여 템플릿을 생성하고 그것을 이용하여 객체를 판별하는 알고리즘을 제안한다.

2. Motion Object Detection

움직이는 물체를 검출하는 방법은 크게 배경

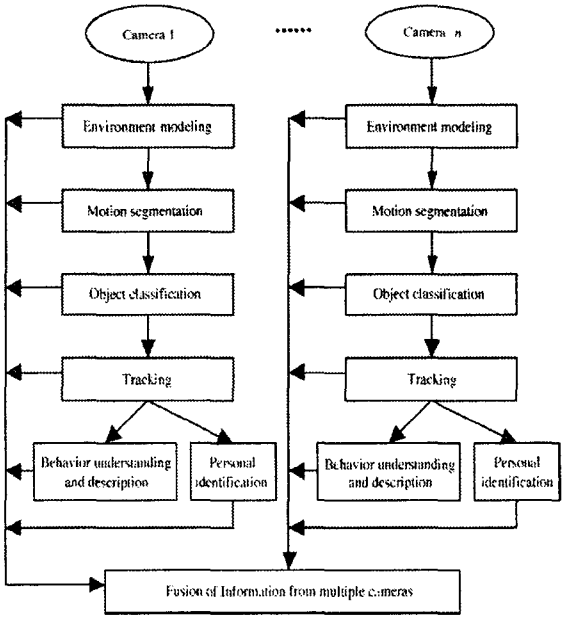


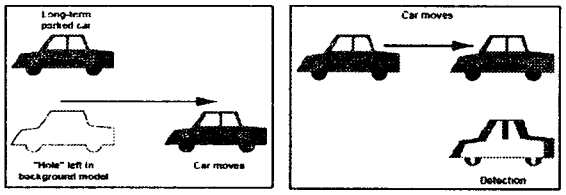
그림 1. 일반적인 영상 추적의 구성도

분리 방법과 프레임간의 차이를 이용하는 방법 (Frame Difference)이 있다. 두 방법은 각각 문제점이 있는데 그림 2에서 보는 것과 같이 배경 추출 방법은 배경으로 분리되어 있던 물

체가 움직일 시 빈 부분이 생겨 움직이는 물체로 오인되는 문제가 있고 프레임간의 차이를 이용하는 방법은 명암의 차이가 주로 외각에 분포되어 있어서 중심 부분이 잘 검출되지 않는 문제가 있다. 이를 해결하기 위하여 두 방법을 결합한 방법을 사용해야 한다[2].

$$M_n(x) = (|I_n(x) - I_{n-1}(x)|) > T_1 \quad (1)$$
$$\text{and}(|I_n(x) - B_n(x)|) > T_2$$

T_1과 T_2는 각각 방법에 따른 임계치를 나타내고 있다. 두 방법을 결합하면 움직이는 부분만 프레임 차이를 이용하여 검출한 후 배경 분리 방법으로 정확한 물체 모양을 검출할 수 있다. 프레임 차이를 이용하는 방법 이용 시 중심 부분을 채우기 위하여 침식과 팽창을 이용하여 넓은 지역을 검출해야 한다.



(a) (b)

그림 2. 움직임 추출 방법의 문제점 (a) 배경 분리 방법의 문제점 (b) 프레임 간 차이를 이용한 방법의 문제점

3. Template 생성

움직이는 물체를 검출한 후 지역 기반 추적(Region Tracker)을 이용하여 각각 객체에 번호를 부여한 후 객체를 추적하게 되는데[3] 겹침 현상이 있는 경우를 대비하여 템플릿을 생성해야 한다. 그림 3은 지역 기반 추적을 이용한 물체 추적의 예를 보여주고 있다.

템플릿을 생성하는 방법도 마찬가지로 배경 분리 방법과 프레임 차이를 이용하는 방법을 사용할 수 있는데 배경 분리 방법은 물체의 윤곽을 잘 추출할 수 있으나 배경의 조도가 변할 시 문제가 발생할 수 있고 프레임 차이를 이용하는 방법은 객체가 천천히 이동할 시 잘 검출하지 못하는 문제점이 있다. 이 문제점을 해결하기 위하여 Adaptive 3-frame difference를 이용한 방법을 사용할 수 있다.

4. 제안하는 알고리즘

4.1 3-Frame Difference

두 프레임 간의 차이를 이용하여 템플릿을 만

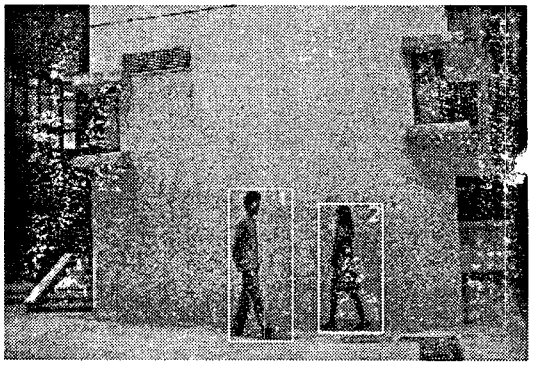


그림 3. 지역 기반 추적

들 경우 전 프레임과 현재 프레임의 객체가 모두 포함되어 정확한 템플릿을 만들 수가 없다. 그러므로 3개의 프레임의 차이를 이용하여 템플릿을 생성하면 정확한 모양의 템플릿을 만들 수 있다.

그림 4에서 표현한 것과 같이 현재 프레임과 이전 2프레임을 이용하여 완전한 모양의 템플릿을 만들 수 있다.

$$R_n(x) = (|I_{n-2}(x) - I_{n-1}(x)|) > T \quad (2)$$
$$\text{and}(|I_{n-1}(x) - I_n(x)|) > T$$

식 (2)에서 표현한 것과 같이 실제로 템플릿이 만들어지는 시기는 n-1번째 프레임이므로 한 프레임 지연되는 것을 알 수 있다.

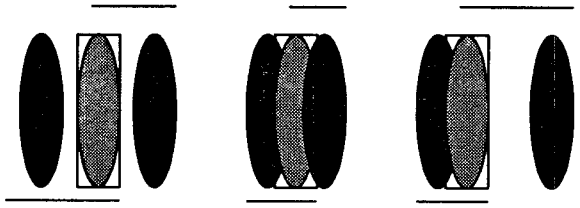


그림 4. 3-frame difference 도식화

4.2 Adaptive 3-Frame Difference

4.1절에서 설명한 3-Frame Difference 방법을 사용하면 정확한 객체의 템플릿을 생성할 수 있지만 천천히 움직이는 객체의 경우는 정확히 검출할 수 없는 문제가 있다. 그러므로 이를 해결하기 위하여 연속된 3 프레임이 아닌 완전한 형태의 템플릿이 생성될 수 있는 3 프레임을 이용하여 프레임간의 차이를 이용하면 정확한 템플릿을 구성할 수 있다. 그림 5는 이를 도식화하여 보여주고 있다.

현재 프레임의 $I_n(x)$과 이와 전혀 겹치지 않는 프레임 $I_{n-1}(x)$을 찾고 두 프레임의 중간에 위치한 $I_{n-1}(x)$을 $I_n(x)$과 $I_{n-1}(x)$의 차

이를 이용하여 템플릿을 생성할 수 있다.

$$R_n(x) = (|I_{n-l}(x) - I_{\frac{2n-l}{2}}(x)|) > T \quad (3)$$
$$\text{and}(|I_{\frac{2n-l}{2}}(x) - I_n(x)|) > T$$

빠른 속도로 움직이는 물체도 마찬가지로 겹치지 않는 프레임을 찾고 프레임간의 차이를 구하면 정확한 템플릿을 생성할 수 있기 때문에 속도가 느린 객체나 속도가 빠른 객체를 구분할 필요가 없다.

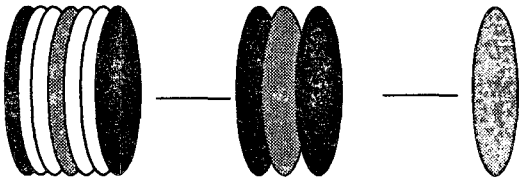
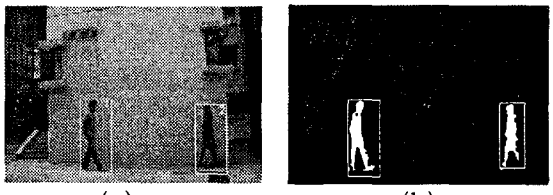


그림 5. Adaptive Frame Difference의 도식화

5. Template Matching

두 객체간의 겹침 현상이 발생할 경우 단순 지역 정보만 이용하여 객체를 판단하면 두 객체를 하나로 인식하거나 새로운 물체로 판단할 수 있다. 그러므로 겹쳐진 상황에서 두 객체를 정확히 판단하기 위하여 템플릿 매칭 알고리즘(Template Matching Algorithm)을 이용해야 한다. 하지만 템플릿 매칭만으로는 정확한 판별이 어려우므로 색 정보를 이용하여 판단하는 방법을 병행하여야 한다[5]. 그림 6은 제안한 알고리즘으로 템플릿을 구성한 것을 보여주고 있다.



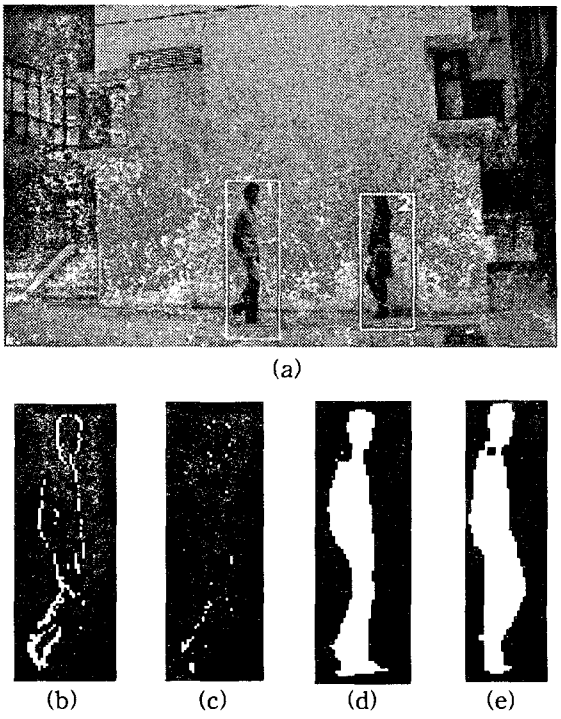
(a) (b)

그림 6. 제안한 알고리즘으로 템플릿을 생성
(a) 객체 추적 상황 (b) 객체의 템플릿 표현

6. 실험

6.1 다양한 방법의 템플릿 생성

느린 속도로 움직이는 사람의 움직임을 추적하는 상황에서 각각 다른 방법으로 템플릿을 생성한 것을 비교해 보면, 2-Frame Difference를 이용한 방법과 3-Frame Difference를 이용한 방법은 움직임 영역의 중심 부분이 잘 검출되지 않기 때문에 완전한 형상의 템플릿을 만들 수 없다. 그러나 배경 분리 방법이나 제안한 알고리즘을 이용한 방법은 완전한 형태의 템플릿을 만드는 것을 볼 수 있다. 그림 7은 각각 다른 방법으로 1번 객체의 템플릿을 생성한 것을 비교한 것이다.



(a)

(b) (c) (d) (e)

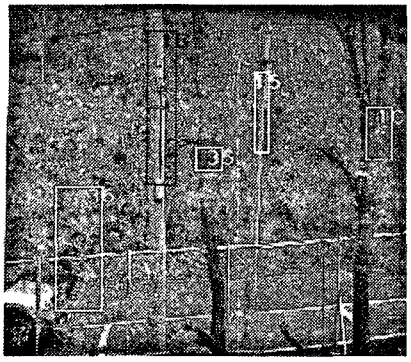
그림 7. 각각 다른 방법을 사용한 템플릿 생성
(a) 객체 추적 상황 (b) 2-Frame Difference 방법
(c) 3-Frame Difference 방법 (d)배경 분리 방법
(e) 제안한 알고리즘을 사용한 방법

6.2 조명 변화 환경에서의 템플릿 생성

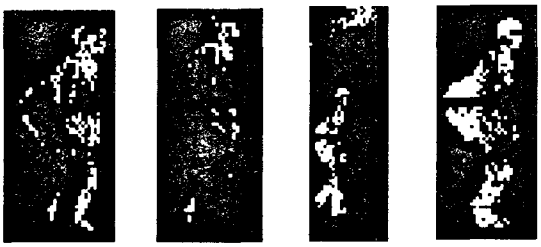
배경의 조명 변화가 없는 상황에서는 배경 분리 방법과 제안한 알고리즘의 차이가 거의 없다. 하지만 배경의 조명이 변하는 상황에서는 제안한 알고리즘이 배경 분리 방법보다 더 나은 성능을 보여주는 것을 확인할 수 있다. 배경의 조명 변화에서도 2-Frame Difference 방법이나 3-Frame Difference 방법은 객체의 중심 부분이 잘 검출되지 않기 때문에 완전한 형상의 템플릿을 만들 수 없다. 그림 8은 조명 변화가 있는 상황에서 36번 객체의 템플릿을 생성한 것을 비교한 것이다.

6. 결과 및 향후 계획

본 논문에서 Adaptive 3-Frame Difference 방법을 이용하여 움직이는 객체의 템플릿을 만



(a)



(b) (c) (d) (e)

그림 8. 각각 다른 방법을 사용한 템플릿 생성
(a) 객체 추적 상황 (b) 2-Frame Difference 방법
(c) 3-Frame Difference 방법 (d) 배경 분리 방법
(e) 제안한 알고리즘을 사용한 방법

드는 방법에 대하여 제안하였다. 객체가 느리게 움직이는 상황이나 배경의 조명 변화가 있는 상황에서 템플릿을 만드는 방법 중 본 논문에서 제안한 Adaptive 3-Frame Difference를 이용한 방법이 가장 좋은 결과를 보여주고 있는 것을 확인 할 수 있다.

그러나, 조명 변화에 의하여 실제로 움직이는 물체가 아닌 경우에도 객체로 잡는 문제가 발생한 것을 확인할 수 있다. 그러므로 배경을 업데이트하는 방법을 사용하여 실제로 움직이는 물체만 객체로 추출해야만 한다. 그리고 템플릿을 생성하기 위하여 겹치지 않는 다음 프레임을 찾는 과정에서 움직이는 물체가 정지하는 경우 무한히 프레임을 저장하는 문제가 발생할 수 있는데 이는 일정 프레임 이하로 프레임을 저장하는 공간을 제안하고 움직임을 멈춘 물체는 레이어(layer)로 분류하여 배경과 함께 움직임이 없는 물체로 간주하는 방법을 사용해야 한다.

참 고 문 헌

[1] Weiming Hu, Tieniu Tan, Liang Wang, Maybank. S, "A survey on visual surveillance of object motion and behaviors", in Systems, Man and Cybernetics, Part C, IEEE Transactions on , 2004, pp. 334 - 352

[2] Robert T. Collins, Alan J. Lipton, Takeo Kanade, Hironobu Fujiyoshi, David Duggins, Yanghai Tsin, David Tolliver, Nobuyoshi Enomoto, Osamu Hasegawa, Peter Burt and Lambert Wixson, "A System for Video Surveillance and Monitoring", tech. report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, May, 2000

[3] Nils T Siebel, "Design and implementation of people tracking algorithms for visual surveillance applications", Reading Univ. March 2003

[4] I. Haritaoglu, D. Harwood, and L. S. Davis, "W^4 : Real-time surveillance of people and their activities," IEEE Trans. Pattern Anal. Machine.

[5] Gary R. Bradski "Computer Vision Face Tracking For Use in a Perceptual User Interface", Intel Technology Journal, 1998, Microcomputer Research Lab, Santa Clara, CA, Intel Corporation.